REMARKS

Status of the Claims

- Claims 1-6 and 8-23 are pending in the Application.
- Claims 1-6 and 8-23 are rejected by Examiner.
- Claims 1, 11, 17, and 21 are amended.

Claim Rejections Pursuant to 35 U.S.C. §102

Examiner has rejected Claims 1-6 and 8-23 under 35 U.S.C. §102(e) as being anticipated by U.S. Pat. No. 7,120,645 to Manikutty et al. (Manikutty). Applicant respectfully traverse the §102(e) rejection.

Applicant amends Claims 1, 11, and 21 to include the aspect that the intermediate language representation includes a tuple operator having three child nodes, the child nodes comprising a list of iterators that construct tuple space, a clause that filters the tuple space, and a clause that produces an outcome of the tuple space, wherein each iterator in the semantic representation of a tuple node corresponds to one column in the tuple space. Applicant finds support for this element in paragraphs 0058 and 0064 of the as-filed specification. Generally, the tuple space operator is described in paragraphs 0058 through 0075 with examples of the intermediate language representation (QIL) in paragraphs 0067 and 0070.

Applicant amends Claims 11, 17, and 21 to include the aspect that utilization of the intermediate language representation in the semantics interpreter, when used in a compiler system having M front-front end languages and N back-end search engines, has a complexity of M plus N compiler implementations. Applicant finds support for this amendment in paragraph 0023 of the as-filed specification.

Applicant amends Claim 21 to include the aspect that the semantic representation is executed directly by one execution engine, and is translated to SQL before execution by a second execution engine, and is executed partially in a third execution engine wherein a balance of the semantic representation is executed in a fourth execution engine. Applicant finds support for this amendment in paragraphs 0040 and 0052 of the as-filed specification.

Manikutty at col. 5, lines 13-25 teaches a technique for executing database commands that involve operations on XML constructs includes receiving the database command. It is then determined whether an XML component operation in the database command can be transformed to a relational database operation, which operates on a particular set of one or more relational database constructs, and which does not involve the XML component operation. If it is determined that the XML operation can be transformed, then the XML component operation is rewritten to a particular relational database operation that does not involve the XML component operation. The particular relational database operation on the particular set of one or more relational database constructs is evaluated. (see col. 5, lines 13-25).

Figure 3 of Manikutty is a flow diagram that illustrates a method of rewriting a database query that has an XML operation. (see col. 4, lines 52-54). Figure 4 of Manikutty is a flow diagram of step 320 of Figure 3. With regard to Step 450 of Figure 4, Manikutty teaches at col. 20 line 61- col. 21, line 17:

"Tree of Canonical XML Generation Functions

> In step 450, canonical XML generation functions are expanded to a normalized tree of canonical functions. In other embodiments, the primitive XML generation operations are expanded to a normalized tree of primitive operations. In a normalized tree, each XEL function has a single non-XML type operand, such as a single scalar operand, or a single abstract data type (ADT) operand, or a single collection type column (typically at a leaf node of the normalized tree), or a set of operands that are of XML type (typically at a parent node of the normalized tree). In some embodiments, a set of operands of XML type might be the result of a tree of canonical XML generation functions. It is ensured that each child in the normalized tree is explicitly tagged. If the child is an ADT that is not tagged, the XEL function is converted to the form XEL (null, "<value expression>" as "<type name>"). Scalar operands are always tagged. For example, FIG. 5 is a block diagram that illustrates a normalized tree 510 of XML generation operations in the sub-query for view dept_xv. In tree 510, each XEL function has a single non-XML type operand, or a set of XML type operands."

(col. 20, line 61 through col. 21, line 17).

Thus, Applicant notes that Manikutty relies on a tree structure to map in canonical XML generation functions. Applicant also notes that Manikutty is absent a teaching of the use of a graph structure having the aspect that the intermediate language representation includes a tuple operator having three child nodes, the child nodes comprising a list of iterators that construct tuple space, a clause that filters the tuple space, and a clause that produces an outcome of the tuple space, wherein each iterator in the semantic representation of a tuple node corresponds to one column in the tuple space as recited in amended Claims 1, 11, and 21.

In addition, Manikutty is absent a teaching that the aspect that utilization of the intermediate language representation in the semantics interpreter, when used in a compiler system having M front-front end languages and N back-end search engines, has a complexity of M plus N compiler implementations as recited in amended Claims 11, 17, and 21.

Also, Manikutty is absent a teaching that the aspect that the semantic representation is executed directly by one execution engine, and is translated to SQL before execution by a second execution engine, and is executed partially in a third execution engine wherein a balance of the semantic representation is executed in a fourth execution engine as recited in amended Claim 21.

Accordingly, Manikutty fails to teach every element of the pending amended independent Claims 1, 11, 17, and 21. Since Manikutty fails to teach all of the elements of the pending claims, then Manikutty cannot anticipate amended independent Claims 1, 11, 17, and 21 and their respective dependent claims.

Applicant therefore respectfully requests withdrawal of the 35 USC §102(e) rejection and submits that Claims 1-6, and 8-23 patentably define over the cited art because all elements of the independent Claims 1, 11, 17 and 21 are not found in the cited art.

Conclusion

Applicant respectfully requests reconsideration and continued examination of all pending claims in light of the amendment and discussion above. Applicant respectfully submits that all pending claims patentably define over the cited art and respectfully requests a Notice of Allowance for all pending claims.

Respectfully Submitted,

Date: October 31, 2007

/Jerome G. Schaefer/

Jerome G. Schaefer
Registration No. 50,800

Woodcock Washburn LLP
Cira Centre
2929 Arch Street, 12th Floor
Philadelphia, PA 19104-2891
Telephone: (215) 568-3100
Facsimile: (215) 568-3439